**哈尔滨工业大学**

**编译原理 2024春**

**实验一**

|  |  |
| --- | --- |
| 学院： | 计算学部 |
| 姓名： | 徐柯炎 |
| 学号： | 2021110683 |
| 指导教师： | 单丽莉 |

# 实验目的

* 巩固对词法分析与语法分析的基本功能和原理的认识。
* 能够应用自动机的知识进行词法分析与语法分析。
* 理解并处理词法分析与语法分析中的异常和错误。

# 实验环境

* GNU Linux Release: Ubuntu 12.04, kernel version 3.2.0–29。
* GCC version 4.6.3。
* GNU Flex version 2.5.35。
* GNU Bison version 2.5。

# 实验内容

编写一个程序对使用C--语言书写的源代码进行词法和语法分析，并打印分析结果。实验要求使用词法分析工具GNU Flex和语法分析工具GNU Bison，并使用C语言来完成。

程序要能够查出C--源代码中可能包含的下述几类错误：

* 词法错误（错误类型A）：出现C--词法中未定义的字符以及任何不符合C--词法单元定义的字符；
* 语法错误（错误类型B）。

# 实验过程

## 程序功能

### 完成了抽象语法树的简单实现

首先我实现了抽象语法树的结构体，如图所示：

其中每一行的功能分别为：

1. int line：当前语法单元的行号；
2. char\* name：语法单元的名称；
3. int n：语法单元子结点的个数；
4. struct TREE \*child[maxNum]：子节点指针数组；
5. union：联合体，用来存放终结节点的数值。

接下来的所有程序的实现都依赖于这个数据结构，当词法分析和语法分析完成后，将采用先序遍历的方法来打印整个语法分析树。

### 词法分析

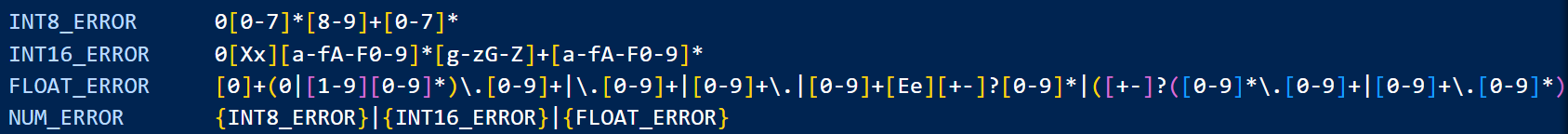
词法分析是语法分析的基础，关于词法分析，我实现了以下功能：

1. 能够正确识别十进制数、八进制数、十六进制数并能进行进制转换；
2. 能够正确识别浮点数；
3. 能够正确识别各类标识符、关键字和标点符号。

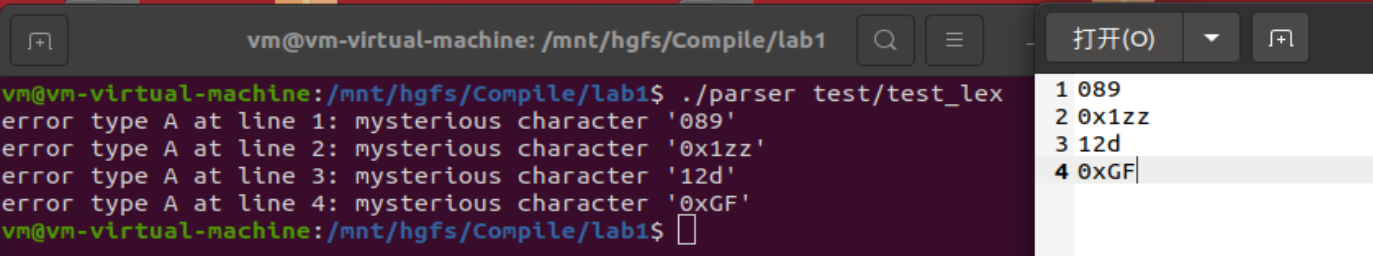
特别地：

1. 能够正确识指数形式的浮点数；
2. 可以判断出错误的八进制数、十六进制数、浮点数、指数形式数。
3. 能够识别所有的注释符号，并判断出错误的注释符号。

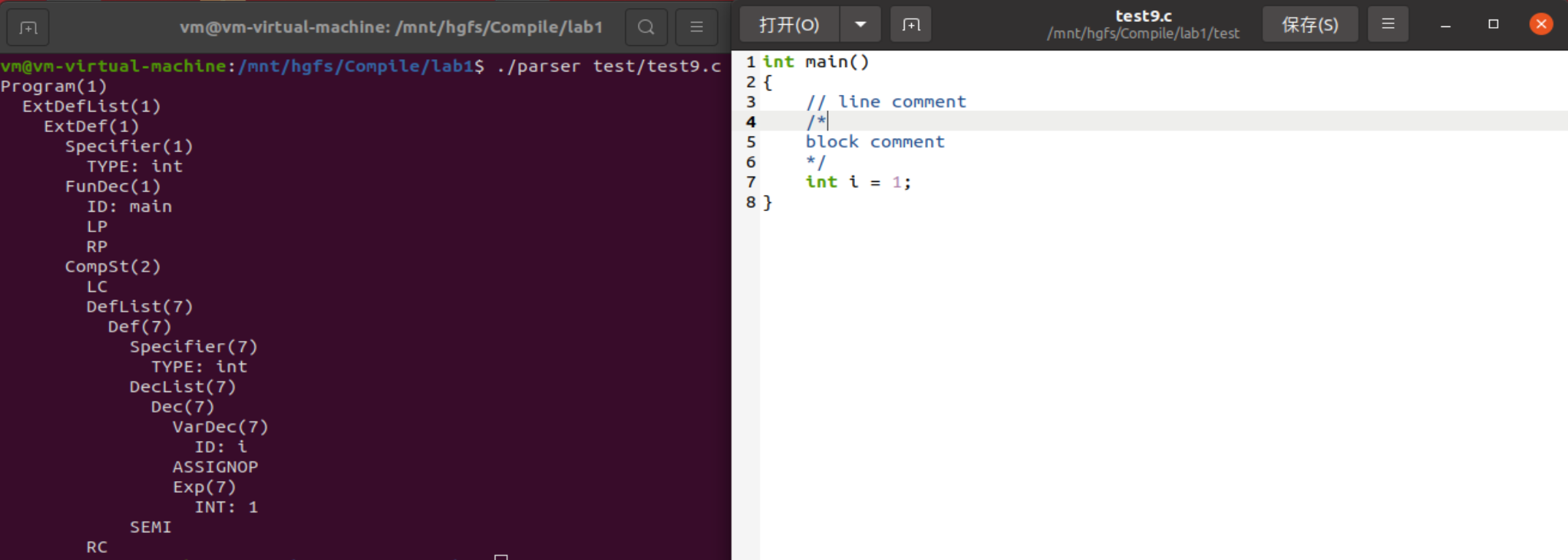
识别错误的各类数字代码如下图所示：



测试代码以及测试结果如下图所示：



本程序还能正确识别注释符号，这里单行的注释采用正则表达式实现，而多行注释采用如下方法：当遇到’/\*’时，执行read\_comment()函数，贪婪匹配’\*/’。测试选做第五个样例如下图所示：



可以发现能正确打印语法树，

### 语法分析

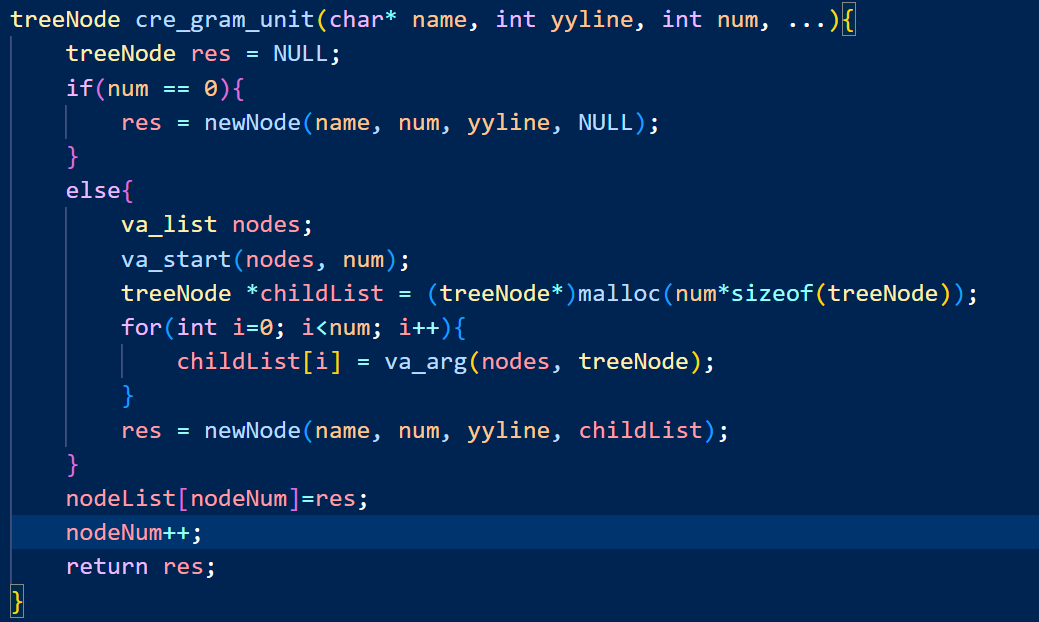
关于语法分析，我实现了以下功能：

1. 使用CFG正确构建抽象语法树；
2. 使用先序遍历正确打印抽象语法树的各个节点以及数值。

特别地：

1. 实现较高层次的模块化，简化代码，增加代码的可读性。

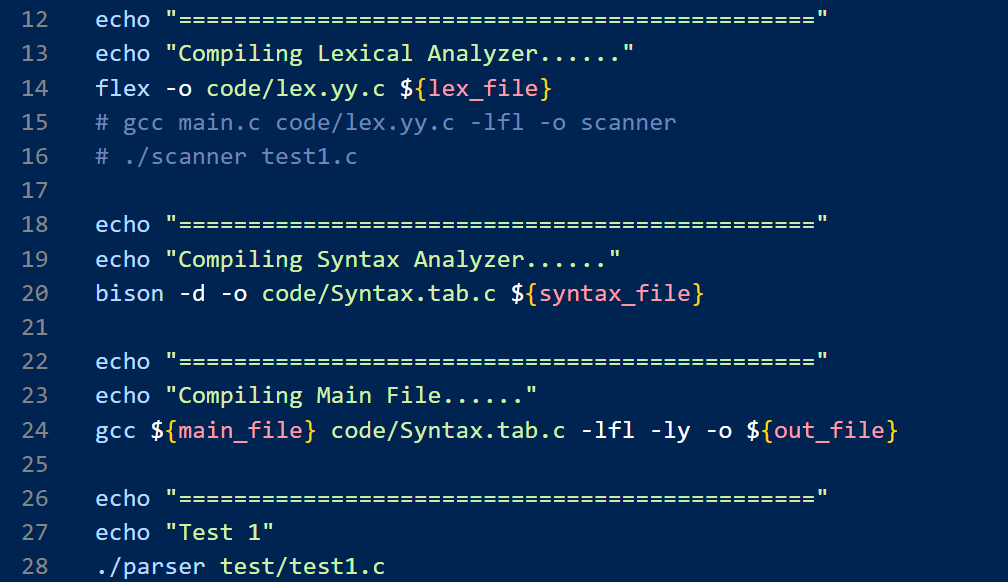
具体来说，我实现了一个可传入可变数量的参数的函数，此函数用来构建语法单元节点，如下图所示：



此函数通过va\_list来传入num个可选参数，也就是这个语法树的子节点，然后循环读入各个子节点，和父节点建立联系，从而实现此语法单元节点的构建。

## 编译过程

这里使用shell脚本来编译我们的flex、bison以及c语言的代码，并且实现测试代码的运行，如下图所示：



其中实现程序的代码放在code文件夹中，测试代码放在test文件夹中，最后编译生成程序parser，通过运行此程序完成测试。

# 实验总结

1. 学会了如何使用flex和bison编写程序分析词法和语法；
2. 复习了词法分析和语法分析的过程；